Generating Definite Descriptions Non-Incrementality, Inference, and Data

Claire Gardent, Hélène Manuélian, Kristina Striegnitz, and Marilisa Amoia

1. Introduction

The generation of referring expressions is a central task for systems that automatically generate natural language texts. Indeed, the correct use of natural language referential devices is crucial for generating successful utterances, i.e., utterances that are easily and correctly understood by the hearer, because referring expressions play an important role in linking an utterance to the previous discourse, the non-linguistic situation the utterance is produced in, and the knowledge of speaker and hearer.

One algorithm that has been particularly influential in this field is the incremental algorithm for generating referring expressions presented in (Dale and Reiter 1995). In this paper, we both extend this basic algorithm to deal with more complex, inference based, definite descriptions and propose an alternative, non-incremental algorithm which circumvents the shortcomings arising from incrementality. Moreover, we present the results of a corpus study on definite descriptions in French which suggest some research directions that could help both widening the range of expressions that can be generated and improving the form and content of generated definite descriptions.

We start (Section 2) with a brief presentation of Dale and Reiter's (1995) incremental algorithm. In Section 3, we turn to the generation of bridging descriptions. Bridging descriptions are definite descriptions that refer to entities which are new to the discourse, but can be linked through world knowledge to an entity that has been mentioned before. We look at the contextual reasoning necessary for generating such definite descriptions and then integrate it into a variant of Dale and Reiter's basic algorithm. In Section 4 we present results of a corpus study which examines the relations that link bridging descriptions to the context. The results offer valuable insights on what kind of information has to be provided for the generation of bridging descriptions. Section 5 looks at another aspect of Dale and Reiter's

algorithm and algorithms derived from it: incrementality. We identify a number of problems that arise from incrementality and present an alternative, non-incremental approach. Section 6, finally, summarises our conclusions and points out open questions that further research on the generation of referring expressions needs to address.

2. Dale and Reiter's incremental algorithm

The algorithm described in (Dale and Reiter 1995) provides the basis for many of the later approaches to the generation of referring expressions (Horacek 1997; Krahmer and Theune 2001; van Deemter 2002). As we will also build on it, we will now sketch the way it works. The input to this algorithm are

- the *context*: a set *C* of positive literals associating entities with relations of arbitrary arity such as shown in Figure 1,
- the *target entity*: the entity *t* which needs to be referred to.

$\{rabbit(r_1),\ rabbit(r_2),\ rabbit(r_3),\ hat(h_1),\ hat(h_2),\ bathtub(b_1),\ white(r_1),\ black(r_2),\ white(r_3),\ in(r_1, h_1),\ in(r_2, h_2),\ in(r_3,b_1)\}$

Figure 1. Representation of a context for Dale and Reiter's incremental algorithm.

The task of the algorithm is to find a *distinguishing description* for the target entity, i.e., a subset *L* of *C* which uniquely identifies the target. In other words, given the context, *L* should give so much information about the target entity that it cannot be confused with any other entity mentioned in *C*. For example, if, given the context of Figure 1, r_2 is the target entity, $\{rabbit(r_2),\ black(r_2)\}$ would be a solution, as there are no other entities which are rabbits and black in the context. However, if r_1 is the target entity, $L=\{rabbit(r_1),\ white(r_1)\}$ would not be sufficient, because there is one *distractor*, i.e., one other entity which also fits the description given by *L*, namely entity r_3. $L=\{rabbit(r_1),\ white(r_1),\ in(r_1,h_1),\ hat(h_1)\}$ would be a solution in this case, because r_1 is the only entity in *C*, which is a white rabbit and is in a hat. Formally, this can be captured as follows: *L* is uniquely identifying the target entity *t*, if $D(t,L,C) = \{t\}$ with $D(t,L,C)$ the set of objects satisfying the description or more formally:

$$D(t,L,C) = \{o| \exists \text{ substitution } s \text{ such that } s(t)=o \text{ and } s(L) \subseteq C\}.$$

The algorithm builds such a distinguishing description *incrementally*. That is, it starts with $L=\varnothing$, adds one property and checks whether this already suffices to distinguish the target from all other entities in the context. If not, it adds the next property and checks again. The algorithm proceeds in this way until either the description uniquely identifies the target or all properties are used up. In this latter case, the algorithm fails as no distinguishing description can be built.

```
initialise
1.    targets ← <t>
2.    L ← ∅
main loop: until no more entities are left on the target list
3.    while targets ≠ <> do
4.        o ← targets[1]
if the first entity on the target list is uniquely identified
5.        if D(o,L,C) = {o}
then take it from the list
6.        then
7.            targets ← targets[2,length(targets)]
else try to extend the description
8.        else
9.            Ps = applicable_literals(L,C)
10.           if Ps = ∅ then return fail
11.           p = select_one(Ps)
12.           update targets and L with p
13.       endif
14.   endwhile
15.   return L
```

Figure 2. Dale and Reiter's incremental algorithm.

The pseudo-code of the algorithm is given in Figure 2. In the beginning, $L=\varnothing$ and the ordered list for keeping track of which entities have to be described contains only the target. Whenever a literal that involves other entities than the target entity is added to the description, these entites are added to the target list. The algorithm then has to make sure that the

description uniquely identifies these entities as well as the initial target entity.

The main loop of the algorithm is controlled by this target list. It stops when the list is empty. Otherwise it takes the first element of the list and checks whether the current *L* uniquely identifies it. If so, that target is removed from the list. Otherwise, *L* has to be extended. If that is not possible because there are no *applicable literals* left in *C*, the algorithm fails. *Applicable literals* are not yet part of *L*, they mention the target, and there is at least one distractor to which this literal doesn't apply. If there are several applicable literals, the algorithm has to choose one. The incremental algorithm presented in (Dale and Reiter 1995) assumes a domain dependent ordering of properties which determines this choice. Other variants of the algorithm might use slightly different mechanisms.

The table in Figure 3 illustrates how the algorithm that we just described would incrementally build a description for entity r_1 given the context in Figure 1. For the purpose of this example, we assume that type properties are chosen before colour properties, which are chosen before locations.

targets	L	distractors	notes
$<r_1>$	$\varnothing$	all entities	extend *L*
$<r_1>$	$\{rabbit(r_1)\}$	$\{r_1, r_2,, r_3\}$	extend *L*
$<r_1>$	$\{rabbit(r_1), white(r_1)\}$	$\{r_1, r_3\}$	extend *L*
$<h_1, r1>$	$\{rabbit(r_1), white(r_1), in(r_1,h_1)\}$	$\{h_1, b_1\}$	extend *L*
$<h_1, r1>$	$\{rabbit(r_1), white(r_1), in(r_1,h_1), hat(h_1)\}$	$\{h_1\}$	h_1 unique
$<r_1>$	$\{rabbit(r_1), white(r_1), in(r_1,h_1), hat(h_1)\}$	$\{r_1\}$	r_1 unique
<>	$\{rabbit(r_1), white(r_1), in(r_1,h_1), hat(h_1)\}$		return *L*

Figure 3. An example run of the incremental algorithm.

3. Definite descriptions and inference

The algorithm outlined in the previous section represents the context as a set of atomic facts. The definite descriptions it generates all refer to entities explicitly mentioned in this context and only contain information explicitly given in this context. In this section, we want to look at an extension of this base algorithm that uses a richer notion of context and can therefore generate definite descriptions which leave implicit how exactly the referent is linked

to the context. We will first present the data that we want to capture and then describe the extended algorithm.

3.1. Inference based definite descriptions

The prototypical use of definite descriptions is as in the following example.

(1) *A woman came in. **The woman** was wearing a beautiful hat.*

In this example, the definite description is referring to an entity that has explicitly been mentioned before and is only using properties that have explicit been attributed to that entity. This kind of definite descriptions is also known as *directly co-referring definite descriptions*. However, corpus studies have shown that this prototypical use of definite descriptions only accounts for about 30% of all definite descriptions that are found in natural text (Poesio and Vieira 1998). Poesio and Vieira found that additional 24% were what we call *inference based definite descriptions*. We distinguish two types: *indirectly co-referring definite descriptions* and *bridging descriptions*. The following is an example of an indirectly co-referring use.

(2) *An actress entered the stage. **The woman** was wearing a big hat.*

The definite NP *the woman* is referring to an entity that has explicitly been mentioned before but uses information that has not explicitly been attributed to that entity before. In (2) this information is a *generalisation* of what is already known about the referent, and hence, could be inferred from the context if some background information of the form '*actresses are women*' was known. (3) shows examples where this is not the case.

(3) a. *John has bought a new car. **The Volvo** delights him.*
b. *I met a man yesterday. **The bastard** stole my money.*

In (3a), the descriptive content of the definite description is a *specification* of what's already known about the referent (Volvos are kinds of cars). In (3b), the definite description is giving a *re-description* of its referent; it adds information which is not in any relation with what's already known about the referent. In the following, we will concentrate on indirectly referring definite descriptions of the type shown in (2).

The second type of inference based definite descriptions, bridging descriptions, are definite descriptions that refer to an entity that has not explicitly been mentioned before. However, it is related to an entity that has been mentioned before. Here is an example of such a case.

(4) *John entered the room. **The ceiling** was very high.*

The ceiling mentioned in the second sentence is a part of the room mentioned in the first sentence. Moreover, our general knowledge about rooms tells us that every room has a ceiling. So, when hearing that there is a room, we already know that there also must be a ceiling. (5a) shows a bridging description that refers to an entity of which we don't know before that it exists. Not all rooms have windows. Therefore, the existence of the window doesn't follow from the context. However, it is quite probable in the given context, as most rooms have windows. This is not the case in (5b). The fact that there is a chandelier neither follows from the context nor is it probable in the given context. But even as it is not common nowadays for rooms to have chandeliers, it is plausible to link the chandelier to the room as rooms typically have lamps and a chandelier is a type of lamp; moreover, it is a type of lamp that usually is found in rooms.

(5) a. *John entered the room. **The windows** looked out to the bay.*
b. *John entered the room. **The chandelier** was sparkling brightly.*

Before we go on, we have to introduce some more terminology. In the rest of this section, we will follow Dale and Reiter and call the referent of the definite description that we are going to generate the *target*. The entity in the discourse context that the target is linked to is called *anchor*. This link between target and anchor is established via the identity relation in the case of co-reference and via some bridging relation in the case of bridging descriptions. Clark (1977), who introduced the term *bridging*, identified various different types of bridging relations, such as the part-of relation, semantic roles of verbs, reasons, consequences. For the moment, we will use a fairly general part-of relation as our only bridging relation. We take this relation to subsume all meronymic relations including functional parts (Kleiber 1997); e.g., presidents are (functional) parts of companies. See

Section 3 for results of a corpus study aiming at a better understanding of what kinds of relations can act as bridging relations.

3.2. An algorithm for generating inference based definite descriptions

We will now see how the basic algorithm of Section 2 can be extended with the necessary knowledge and reasoning to generate inference based definite descriptions. First, we will determine what kind of information sources are involved in the generation of these definite descriptions and re-specify the representation of the context accordingly. Then, we will characterise the conditions under which a definite description can be thought to be uniquely identifying when taking into account this additional knowledge and reasoning. Finally, we integrate these conditions into the incremental algorithm and illustrate the workings of the extended algorithm by going through an example.

3.2.1. The discourse context

In the incremental algorithm, the context is a list of positive literals recording information that is known to both the hearer and the speaker. It specifies entities and lists their properties. To deal with bridging definites and indirectly anaphoric definites, this notion of context has to be extended in two ways. First, we need a model of the relevant world and lexical knowledge containing information of the type: *restaurants have cooks, poodles are pets*. Second, we have to be able to access knowledge that the speaker might have about the situation and which is not shared by the hearer. The definite description in (5a), for instance, refers to an entity which the hearer does not know about before hearing the sentence. For generation we have to model this fact. Our context will therefore consist of three parts now:

Discourse Model (DM): A list of positive literals modelling the previous discourse. This is essentially the context used in the incremental algorithm. Entities which are mentioned in the discourse model are called *discourse old* entities.
E.g.: *DM = {restaurant(r), Italian(r), …}*

Background Knowledge (BKL): Lexical and world knowledge shared by the speaker and the hearer. BKL together with the DM is the *shared knowledge (ShKL)*, i.e., the information that both the speaker and the hearer know about.
E.g.: $BKL = \{\forall x\ (restaurant(x) \rightarrow \exists y\ (cook(y) \wedge part\text{-}of(y,x))), \ldots\}$

Speaker Model (SM): Additional knowledge of the speaker.
E.g.: $DM = \{cook(c), part\text{-}of(c,r), \ldots\}$

3.2.2. Intended and potential anchors

Now, that we have the enriched representation of the context, we have to examine how it interacts with the requirements under which a definite description can be used. In other words, what does it mean now for an entity to be uniquely identified by a definite description? To answer this question, we employ the terms *intended anchors* and *potential anchors*. Intended anchors (IA) are those discourse old entities which the speaker intends to act as the anchor of the target, and potential anchors (PA) are those entities which from the hearer's point of view (i.e., taking into account only shared knowledge) could act as anchors of the target.

In the basic algorithm, the only intended anchor is always the target itself. With bridging descriptions this definition obviously has to be loosened. (It is, in fact, one of the defining characteristics of bridging descriptions that the anchor is different from the target.) We will say that *intended anchors* are all those discourse old entities *o*, such that either *o* is equal to the target, or it follows from the speaker's knowledge that *o* is related to the target via a bridging relation. For instance: Let *c* be the target. If *c* is mentioned in *DM*, then *c* is an intended anchor. Furthermore, if $SM \models bridge(o,c)$, then *o* is an intended anchor of *c* as well.

The *potential anchors of a target t given the description L in the context C* are given by *D(t,L,C)* in the basic algorithm. Also this definition needs to be loosened now. The potential anchors for the definite description *the cook*, for example, should include not only all the cooks in the discourse model but also all restaurants. Potential anchors will therefore be all discourse old entities *o* that, given the shared knowledge *ShKL* (i.e., *DM* and *BKL* together), fulfill one of the following conditions:

1. *o* fits the description *L* of *t*. For instance: $L = \{cook(c)\}$, $t = c$, and $ShKL \models cook(o)$. This is the case if $cook(o) \in DM$.
2. *o* is related (via a bridging relation) to some entity which fits the description. For instance: $L = \{cook(c)\}$, $t = c$, and $ShKL \models \exists x\ (bridge(x,o) \wedge cook(x))$. This is the case if $restaurant(o) \in DM$ and $\forall x\ (restaurant(x) \rightarrow \exists y\ (cook(y) \wedge part\text{-}of(y,x))) \in BKL$.
3. *o* is related (via a bridging relation) to some entity which fits a generalisation of the description. For instance: $L = \{chandelier(c)\}$, $t = c$, and there is a predicate *G* such that $ShKL \models \exists x\ (bridge(x,o) \wedge G(x))$ and $BKL \models chandelier(c) \rightarrow G(c)$. This is the case if $room(r) \in DM$, and $\forall x\ (room(x) \rightarrow \exists y\ (furniture(y) \wedge part\text{-}of(y,x))) \in BKL$, $\forall x\ (chandelier(x) \rightarrow lamp(x)) \in BKL$, and $\forall x\ (lamp(x) \rightarrow furniture(x)) \in BKL$.

Note that the above conditions don't allow to link potential anchors to the target via *chains* of bridging relations. Here is an example to illustrate why we want to exclude this. Assume that $L = \{cook(c)\}$, $t = c$, $university(u) \in DM$, $canteen(r) \in DM$, $\forall x\ (university(x) \rightarrow \exists y\ (canteen(y) \wedge part\text{-}of(y,x))) \in BKL$, and $\forall x\ (canteen(x) \rightarrow \exists y\ (cook(y) \wedge part\text{-}of(y,x))) \in BKL$. In this situation, the university *u* should not be a potential anchor for target *c* (contrary to what would be predicted if chaining of bridging relations was allowed), since otherwise Example (6) could be produced.

(6) *John went to the university. ???**The cook** wore a white apron.*

Now, that we have defined intended and potential anchors[1], we can look at the role they play in determining whether the use of a definite description is contextually appropriate. As we have just seen, intended anchors are those entities in the discourse model that the speaker wants to link the referent of the definite description to, and potential anchors are those entities that the definite description could possibly be linked to when considering only the hearer's knowledge. A successful definite description therefore has to avoid mismatches between the set of intended anchors and the set of potential anchors. The hearer has to recognise the anchors intended by the speaker as potential anchors, i.e., $IA(t,SM) \subseteq PA(t,L,ShKL)$, and he has to be able to rule out all those entities as anchors which are not intended to be anchors by the speaker, i.e., $PA(t,L,ShKL) \subseteq IA(t,SM)$. The first condition, which we call *familiarity*, ensures that the information which the speaker uses in the description lets the hearer establish the link between target and anchor. The

second condition, which we call *uniqueness of the anchor*, makes sure that the definite description contains enough information to rule out all distracting anchors.

These two conditions explain why the definite description *the cook* is not appropriate in (7a) and (7b): it violates the familiarity condition (zoos don't usually have cooks) and the uniqueness condition (which of the two restaurants is the anchor?), respectively.

(7) a. *John took Jim to the zoo. ??? **The cook** was wearing a white hat.*
b. *There is an Italian restaurant and a Chinese one at the end of the street. ??? **The cook** is very good..*

The two conditions do, however, not explain why *the page* in (8) is inappropriate as well. For cases of this type, we need one more condition saying that the target must be *unique wrt. the anchor*. I.e., it must be coherent with the context to assume that there is only one entity which fits the description and is related to the anchor via a bridging relation.

(8) *John took the book back to the shop. **The page** was missing.*

We conclude this section with a summary of the concepts and conditions that have just been introduced and on which the algorithm in the following section will be based.

inteded anchors (IA(t,SM)): set of discourse old entities that the speaker wants to relate the target to.

potential anchors (PA(t,L,ShKL)): set of discourse old entities that, given only the hearer's knowledge, could act as anchors for the target.

familiarity condition: $IA(t,SM) \subseteq PA(t,L,ShKL)$.

uniqueness condition I: $PA(t,L,ShKL) \subseteq IA(t,SM)$.

uniqueness condition II: It is consistent with the context to assume that the target is the only entity which fits the description and is related to the anchor via a bridging relation.

3.2.3. Extending the incremental algorithm

Now, we will modify the basic algorithm presented in Section 2, so that it works with the extended representation of the context and uses the notions of intended and potential anchors to decide when to terminate successfully and when to fail. The pseudo-code is given in Figure 4.

```
initialise
1.    targets ← <t>
2.    L ← ∅
main loop: until no more entities are left on the target list
3.    while targets ≠ <> do
4.    o ← targets[1]
if the description rules out intended anchors of o
5.        if IA(o,SM) – PA(o,L,ShKL) ≠ ∅
then return and classify L as unfamiliar
6.        then
7.            return <unfamiliar, L>
8.        endif
if o is uniquely identified
9.        if PA(o,L,ShKL) ⊆ IA(o,SM) and o is unique wrt. IA(o,SM)
then take it from the target list
10.       then
11.           targets ← targets[2,length(targets)]
else try to extend the description
12.       else
13.           Ps = applicable_literals(L,C)
14.           if Ps = ∅ then return <non-uniquely-identified, L>
15.           p = select_one(Ps)
16.           update targets and L with p
17.       endif
18.   endwhile
19.   return <uniquely-identified, L>
```

Figure 4. The extended algorithm.

Like the basic algorithm, the extended algorithm starts with $L = \varnothing$. It then adds literals to L until the uniqueness condition is satisfied, while making sure that the familiarity condition is not violated. There are two ways in

which this algorithm can fail to build a uniquely identifying description. In one case, there are no more applicable literals which could be added to *L* but the uniqueness condition is not yet satisfied. This is similar to the situation in which the basic algorithm fails. The other case is new: the algorithm fails if no description preserving the familiarity condition can be built. Applicable literals are computed as in the basic algorithm, but we furthermore give preference to literals that preserve the familiarity condition.

We will now go through an example to illustrate how the extended algorithm works. Assume that the task is to generate an expression referring to entity *c* given the following context:

BKL: $\forall x\,(restaurant(x) \rightarrow \exists y\,(cook(y) \wedge part\text{-}of(y,x)))$
DM: $restaurant(r_1)$, $italian(r_1)$, $restaurant(r_2)$, $chinese(r_2)$
SM: $cook(c)$, $part\text{-}of(c,r_1)$

That is, there are two restaurants in the context, an Italian one and a Chinese one, and the speaker wants to refer to the cook of the Italian one, who has not been mentioned before. The table in Figure 5 shows how the description and the status of the familiarity and the uniqueness condition evolve.

targets	*description*	PA	IA	
$<c>$	$\varnothing$	all entities	$\{r_1\}$	Fam.: ✓ Uniq.: ✗
$<c>$	$\{cook(c)\}$	$\{r_1, r_2\}$	$\{r_1\}$	Fam.: ✓ Uniq.: ✗
$<r_1, c>$	$\{cook(c), part\_of(c, r_1)\}$	$\{r_1, r_2\}$	$\{r_1\}$	Fam.: ✓ Uniq.: ✗
$<r_1, c>$	$\{cook(c), part\_of(c, r_1), restaurant(r_1)\}$	$\{r_1, r_2\}$	$\{r_1\}$	Fam.: ✓ Uniq.: ✗
$<r_1, c>$	$\{cook(c), part\_of(c, r_1), restaurant(r_1), italian(r_1)\}$	$\{r_1\}$	$\{r_1\}$	Fam.: ✓ Uniq.: ✓
$<c>$	$\{cook(c), part\_of(c, r_1), restaurant(r_1), italian(r_1)\}$	$\{r_1\}$	$\{r_1\}$	Fam.: ✓ Uniq.: ✓
$< >$	$\{cook(c), part\_of(c, r_1), restaurant(r_1), italian(r_1)\}$	$\Rightarrow$ description is uniquely identifying		

Figure 5. An example run of the extended algorithm.

3.3. Summary and Discussion

Starting from a tripartite context representation we have defined conditions that govern the generation of inference based definite descriptions. Based on these we have then extended the base algorithm from Section 2.

This algorithm has been implemented using description logic to specify the context and the description logic reasoning system RACER (Haarslev and Möller 2001) to carry out the necessary inferences.

What counts as a bridging relation is a question that is quite central for our algorithm: if the definition is too loose the algorithm will overgenerate, i.e., it will produce bridging descriptions that cannot be linked to the context by a human reader/hearer, and if it is too strict it will undergenerate in the opposite direction. In the following section, we will describe a corpus study which was aimed at getting a better idea of how the relation *bridging relation* should be defined and what kind of knowledge would have to be coded in the context knowledge bases in order to be able to generate bridging descriptions.

What we have described in this section concentrated on the contextual reasoning involved in the generation of inference based descriptions. In particular, we have ignored the fact that bridging descriptions, similar to pronouns, require highly salient anchors (Gundel, Hedberg, and Zacharski 1993). So, to be able to use our algorithm for the generation of referring expressions within larger texts, it would have to be augmented with a representation of salience in order to filter out potential anchors which are no longer accessible. See, e.g., (Krahmer and Theune 2001) for a proposal addressing this question.

4. Corpus study

The algorithm in the previous section requires a knowledge base that specifies which concepts can be linked through bridging relation. When building a system for generating referring expressions in a given domain one therefore has to decide which relations linking concepts of that domain can act as bridging relations and hence have to be included in the knowledge base. To get a better understanding of what relations can be involved in bridging descriptions we conducted a corpus study on French data. In this section, we describe this corpus study and discuss the results.

We start (Section 4.1) with a brief description of the corpus that we used. The annotation was done in two passes. In the first pass, which is described in Section 4.2, we classified all definite descriptions according to their referential status (co-reference, bridging, first mention) in order to identify the bridging descriptions. In the second pass, we annotated through which relations the referents of the bridging descriptions were linked to their anchors. The results of this part of the study are presented in Section 4.3.

4.1. Corpus and annotation method

We used a 65,000 words subcorpus extracted from the French PAROLE corpus[1] (Lecomte 1997). This corpus consists of articles taken from the newspaper *Le Monde* and covers a wide range of topics (sports, culture, politics, economics and leisure). It is annotated at the morpho-syntactic level in accordance with the annotation scheme MULTITAG/MULTEXT of the GRACE project (Beaumont, Lecomte, and Hathout 1998). In particular, each determiner is marked as either definite, indefinite, contracted (i.e., contraction of a preposition and a determiner), partitive, demonstrative, possessive, relative, exclamative or interrogative.

We used Gsearch (Corley et al. 2001) for automatically identifying the definite descriptions and MMAX (Müller and Strube 2001) to support the annotation process.

All annotations that are presented in the following were done and agreed on by the authors.

4.2. Referential status of definite descriptions

The first step of the corpus study aims at identifying bridging descriptions. To this end, we carried out a first annotation pass to classify definite descriptions with respect to their referential status. We use an annotation scheme which is loosely based on the proposals in (Poesio and Vieira 1998) and which distinguishes the following categories.

direct co-reference: A nominal antecedent can be found. It refers to the same entity as the target definite description. The head nouns of the antecedent NP and the target NP are the same.

indirect co-reference: A nominal antecedent can be found. It refers to the same entity as the target definite description. The head

nouns of the antecedent NP and the target NP are the *not* same. Indirect co-reference can be realised by using lexical relations such as hyponymy, hypernymy and synonymy.

bridging:. A nominal or verbal antecedent can be found. It refers to a different entity than the target definite description, but, due to general lexical or world knowledge, the referent of the target NP is interpreted as a part of the referent of the antecedent or as an object linked to the it.

first mention: There is no antecedent in the text that fits one of the previous conditions.

The results of this first annotation pass are given in Figure 6.

direct co-reference	612	6,96%
indirect co-reference	869	9,89%
bridging	416	4,73%
first mention	6892	78,42%
total	8789	100%

Figure 6. Results of the first annotation pass.

As can be seen, the proportion of first mention definites is very high (almost 80%). In comparison, Poesio and Vieira (1998) report a rate of around 50%, and Fraurud (1990) found that 60.9% of the definite descriptions in Swedish text are first mention. There are several factors which might be responsible for this difference. One is that in our definition the first mention class plays, to a certain extent, the role of the "waste paper basked": everything which doesn't fulfil the criteria of any of the other classes gets classified as first mention. In particular, event or discourse deictic anaphora are first mention according to our definition. But there are also factors related to the language and text type that might play a role. First, many country names, institution acronyms and idioms in French involve a definite article (e.g., *la France*, *la Côte d'Ivoire* for country names; *le CNRS*, *la CNCL* for acronyms and *avoir la main* for idioms). Second, we classified repeated use of similar definite descriptions separated from each other by a long distance as first mention. Third, we found a high number of generic uses. Fourth, the corpus contains a very high percentage (19.63%) of containing inferable (i.e., definite descriptions such as *the heat of the sun* which are in fact familiar through their explicit relation to a known entity).

With respect to the generation task, these results indicate that only a very small portion of the data can be handled by the Dale and Reiter algorithm which in essence deals only with direct co-reference cases that is, 6.96% of the cases found in our corpus. Extending the algorithm as is proposed in Section 3 to inference base definite descriptions increases the coverage by 14.6%. Nonetheless, there remain 78.42% of first mention definites which cannot be generated. To properly treat these, the existing algorithms have to be extended (i) to deal with discourse new information in definite noun phrases and (ii) to generate containing inferables.

4.3. Bridging relations

Now, we come to the second part of our corpus study which is aimed at answering the following two questions. 1.) What relations are used to link the referents of bridging descriptions to their anchor? 2.) What knowledge sources provide these relations: is it a lexical relation (e.g., meronymy, hyponymy, synonymy) whose encoding is part of resources such as WordNet? Is it given by world knowledge? Or is it given by a lexicographic definition?

We will first present the typology of bridging relations that we developed based on classifications proposed in the literature and the evidence that we found in our corpus. Then we show the distribution of bridging relations in our corpus wrt. to this typology and discuss the implications those results have for the generation of bridging descriptions.

4.3.1. A typology of bridging relations

The annotation scheme we propose classifies bridging descriptions into five broader classes: *set membership, thematic, definitional, co-participant, non-lexical*. We now discuss each of these in more detail showing, in particular, how a specific relation is identified and how its semantics is established. (Also see (Gardent, Manuélian, and Kow 2003).)

Set membership. This class covers cases where the target is either a member or a subset of the set referred to by the antecedent (a group of similar individuals); e.g., *seminars/the last seminar*. The semantics of this bridging relation is set membership or subset. The anchor must be a set of individuals and the target an individual or a set of individuals.

Thematic. As illustrated by the pair *murder/murderer*, the target can be related to the anchor via a thematic relation (a murderer is the *agent* of a murder). More generally, a thematic bridge links an individual to an event via a thematic relation defined by the thematic grid of the event. As a result, the property denoted by the noun characterising the individual must be subsumed by the characterisation given by the definition of the thematic grid.

Definitional. In this case, the implicit bridging relation holding between anchor and target is given by the dictionary definition of either the target or the anchor. For instance, in the pair *operation/convalescence* a convalescence (the target) can be defined to be the period *following* an operation (the anchor) or a disease so that in this case, the bridging relation between anchor and target is one of temporal succession.

In a definitional bridge, the definition usually imposes a sortal restriction which must be satisfied by the related object (anchor or target). The property declared (in the text) to hold of the related object must thus be subsumed by the property requested to hold of the related object by the definition.

We distinguish three kinds of definitional relations: meronymic relations, relations between individuals and an attribute, and relations between individuals and an associate.

Meronymic relations are relations which can be expressed using constructions with *part of* or *has*. For two objects X and Y to be in a meronymic relation, it must be possible to say that X usually has Y *and* that Y usually is a part of X. The meronymic relation implies (spatial, temporal or abstract) inclusion and can only hold between entities of the same ontological types (individuals, events etc.). Following (Winston, Chaffin, and Hermann 1987), we assume various types of meronymic relations, such as *whole/part*, *whole/piece*, *individual/stuff*, *collection/member*, *place/area*, *event/subevent*; for a more precise definition of each of these relations, we refer the reader to (Winston, Chaffin, and Hermann 1987). Additionally, we assume an *individual/function* meronymic relation (e.g., *a club*/*the president*) which involves a definitional bridge holding between individuals with one of the related individuals being described by his profession or function wrt. the other (Kleiber 1997).

Contrary to the meronymic relations, the two other types of definitional bridging relations do not imply inclusion but a simple implication relation (a teacher implies some audience, a surface implies an object etc.). More specifically, **individual/attribute** pairs (e.g., *a person/the age*) involve a definitional bridge holding between individuals with one of the related

individuals being a feature (i.e., something that takes a value within a given domain). **Individual/associate** pairs, such as *question/answer*, may involve two individual, one individual and an event or two events. Neither of the two is a part of the other, but a dictionary definition will define one in terms of the other.

Co-participants. There are cases where the relation holding between target and anchor is mediated by a third object which is mentioned in the dictionary definitions of both target noun and antecedent. For instance, the pair *trip/seat* is related by the relation "in vehicle used for" which can be reconstructed from the definition of the target ("a seat is a place reserved for sitting in a vehicle or a room") and of the anchor ("a trip is a displacement of persons by some means of transport"). In such cases, the definitions of the target and the anchor involve two properties P_a and P_t which stand in a subsumption relation (here, *vehicle* is subsumed by *means of transport*).

Non lexical. Finally, there are cases such as *Grenoble/the region* or *fight/the dead* where no amount of lexical knowledge will help and where the relation holding between target and anchor is given either by discourse structure (circumstantial) or by our knowledge of the world and of how things work (WKL).

4.3.2. Distribution of bridging relations in the corpus

In a second annotation pass, we classified 359 bridging descriptions according to the typology just presented. The results, given in Figure 7, suggest the following preliminary conclusions.

First, the importance of the meronymic relation, which is often taken to be the canonical example of a bridging relation, has been confirmed: 52% of the bridging descriptions involve this relation. Since, moreover, the meronymy relation is encoded in WordNet, this suggests that many cases of bridging definite descriptions could be processed using WordNet (Fellbaum 1998). We thus did a first manual search through WordNet, checking for each bridging description involving a meronymic relation encoded in WordNet (i.e., whole/part, collection/member and individual/stuff) to check whether it was related by a direct or indirect (i.e., inherited through a hyponym) meronymic link to its anchor. Unfortunately, we found that only 38 of our 187 meronymic cases were present in WordNet. However, a closer look at the data shows that only a relatively small number of different object sorts are involved in meronymic relations in our corpus (town parts, country

parts, enterprise parts, etc.). This suggests that it should be possible, given a domain and sublanguage, to extend WordNet with the meronymic information necessary to process most of the bridging descriptions involving this relation; or alternatively, to develop the appropriate meronymic knowledge.

Class	No. of occurrences	Proportion
set membership	**21**	**5.85%**
thematic	**19**	**5.29%**
definitional	**283**	**78.83%**
individual/attribute	**32**	**8.91%**
individual/associate	**64**	**17.83%**
meronymic relations	**187**	**52.09%**
whole/part	89	24.80%
whole/piece	0	0%
individual/stuff	0	0%
collection/member	22	6.13%
place/area	26	7.24%
event/subevent	16	4.46%
individual/function	34	9.47%
co-participants	**8**	**2.23%**
non lexical	**28**	**7.80%**
circumstantial	17	4.74%
WKL	11	3.06%

Figure 7. Bridging relations.

Second, and again this is important for processing purposes, the number of cases involving non-lexical knowledge is relatively small with 4.7% of the definite descriptions involving a circumstantial relation (i.e., non knowledge based spatial or temporal inclusion e.g., *laguna/the inhabitants*) and 3% involving world knowledge (no lexical relation can be found between anchor and target e.g., *war/survivors*, *fight/dead*). In such cases, the relation between target and anchor has to be found either (in the first case) through discourse structure (the structure of discourse determines in some way the relation between predicates, arguments and modifiers) or (in the second type of cases) through some complex reasoning (a fight can result in a person being hurt; one form of being hurt is to be dead etc.).

Third, an important class of bridging that does not appear in the literature but that turned out to be quantitatively non negligible is the class of individual/associate pairs (17.8%). This class covers cases where the lexicographic definition of the target implies the existence of a target related entity whose sort subsumes the sort of the anchor. The bridging relation in such cases is the relation given by the lexicographic definition (cf. examples as *operation/convalescence*, *athletics/national federation*, *question/answer investigation/witness report*). For computational processing, the individual/associate class is problematic because it presupposes the availability of lexicographic definitions usable computationally.

Finally, the thematic class which represents 5.3% of the found bridging descriptions, could be processed using a tool such as FrameNet (Baker, Fillmore, and Lowe 1998) in which words are associated with a frame (or script) specifying the frame elements (aka thematic roles) likely to participate in the scenario evoked by that frame. A preliminary manual search shows that this is indeed the case – for 14 of the 19 thematic cases, we found a frame corresponding to the anchor and containing the target as a frame element.

In summary, it seems that for the data found in the PAROLE corpus, roughly 65% of bridging definite descriptions could be processed using either FrameNet, WordNet or some limited form of lexical reasoning. The remaining 35% requires either lexicographic definitions (17.8%), essential attribute information (9%), discourse structure information (4.7%) or deep knowledge based reasoning (3%).

5. A non-incremental algorithm for generating definite descriptions

Both algorithms that we have seen so far, the base algorithm of Section 2 as well as the inference based version of Section 3, build the distinguishing description incrementally, i.e., the output is constructed one step at a time and without backtracking. Most extensions to Dale and Reiter's algorithm that have been proposed follow this incremental approach. In particular, its extension by van Deemter (2002) to boolean properties and non singleton sets of individuals is incremental. Similarly, the algorithms described by Horacek (1997) and Stone (1998) interleave the incremental computation of a distinguishing description with an incremental construction of the syntactic tree associated by the grammar with this description. Each property selected to better identify the target set is used to retrieve a lexical entry whose

semantics is this property and this lexical entry is immediately integrated into the current description. Thus, in these *doubly incremental* approaches, both the content and the form of a definite description are determined greedily, i.e., locally and without backtracking.

In this part of the paper, we explore the feasibility and usefulness of developing an alternative *non-incremental* algorithm. We start (Section 5.1) by presenting a non-incremental, constraint-based algorithm for generating distinguishing descriptions and showing how it can be integrated in a surface realisation algorithm. We then show that such a non-incremental algorithm is advantageous in at least two ways: it provides linguistically and cognitively better distinguishing descriptions than the incremental algorithm (Section 5.2); and it provides top-down guidance for the surface realisation algorithm which better supports the realisation of definite descriptions (Section 5.3).

5.1. A constraint-based non-incremental algorithm for generating definite descriptions

As Dale and Reiter (1995) show, the problem of finding minimal distinguishing descriptions can be formulated as a minimal set cover problem and is therefore known to be NP hard (Garey and Johnson 1979).

The alternative algorithm we propose is therefore based on the use of constraint programming (CP), a paradigm aimed at efficiently solving NP hard combinatoric problems. Instead of following a *generate-and-test* strategy which might result in an intractable search space, CP minimises the search space by following a *propagate-and-distribute* strategy where propagation draws inferences on the basis of efficient, deterministic inference rules and distribution, i.e., case distinctions for a variable value, is performed only when necessary because no further propagation steps are possible (Roy and Haridi 2003).

The basic algorithm. Consider the definition of a distinguishing description given in (Dale and Reiter 1995).

> Let r be the intended referent, then a set L of attribute-value pairs is a distinguishing description for r if the following two conditions hold:
> **C1:** Every attribute-value pair in L applies to r: that is, every element of L specifies an attribute value that r possesses.
> **C2:** For every other entity c of the context, there is at least one element l of L that does not apply to c: that is, there is an

> *l* in *L* that specifies an attribute-value that *c* does not possess.
> *l* is said to rule out *c*.

This definition can easily be lifted to the case where the target is not a single entity but a set *S* of entities. In this case, every attribute-value pair in *L* has to apply to all $r \in S$. This definition can be stated in terms of constraints over sets as follows.

> *I*: the universe;
> P^+_x: the set of properties some entity *x* has;
> $P^-_x = P - P^+_x$ the set of properties some entity *x* does not have;
> $P^+_s = \cap_{x \in s} P^+_x$: the set of properties true of all elements of S;
> $P^-_s = P - \cup_{x \in s} P^+_x$ the set of properties false of all elements of S;
> $L = \langle P^+, P^- \rangle$ is a *basic distinguishing description for* S iff:
> 1. $P^+ \subseteq P^+_s$
> 2. $P^- \subseteq P^-_s$ and
> 3. $\forall c \in I–S, \left| (P^+ – P^+_c) \cup (P^- \cap P^+_c) \right| > 0$

A description for the target set *S* is represented by a pair of set variables $\langle P^+, P^- \rangle$ constrained to be a subset of the set of positive properties of *S*, P^+_s (i.e., properties that are true of all elements in *S*), and of the set of negative properties of *S*, P^-_s (i.e., properties that are true of none of the elements in *S*), respectively. The third constraint ensures that the conjunction of properties thus built eliminates all distractors, i.e., each element of the universe which is not in *S*. More specifically, it states that for each distractor *c* there is at least one property *P* such that either *P* is true of (all elements in) *S* but not of *c* or *P* is false of (all elements in) *S* and true of *c*.

If the problem is formulated in this way, it can be solved using a constraint programming language such as Mozart/Oz (Programming Systems Lab 1998) which supports set variables. Every assignment of values to variables which satisfies the constraints in the definition given above is then a possible solution, i.e., a distinguishing description for the given target set in the given context.

Additionally, a *distribution strategy* needs to be made precise which specifies how to search for solutions. We want to ensure that smaller solutions are preferred and therefore distribute (i.e., make case distinctions) over the cardinality of the output description $\left| P^+ \cup P^- \right|$ starting with the lowest possible value. That is, first the algorithm will try to find a description $\langle P^+, P^- \rangle$ with cardinality one, then with cardinality two etc. The

algorithm stops as soon as it finds a solution. In this way, the description output by the algorithm is guaranteed to always be the shortest possible description.

	c_1	c_2	c_3	p_1	b_1
cup	•	•	•		
plate				•	
bowl					•
red	•	•			•
blue			•	•	

Figure 8. An example context.

To illustrate this we will assume the context in Figure 8. Now, let our target set $S = \{c_1, c_2\}$. Then $P^+_s = \{cup, red\}$ and $P^-_s = \{plate, bowl, red\}$. There is no description containing only one negative or positive property that distinguishes the elements of S from all other elements in the universe. So, the algorithm will look for one consisting of two properties and finds the following two possibilities: $L = \langle\{cup, red\}, \varnothing\rangle$ (*the red cups*) or $L = \langle\{cup\}, \{blue\}\rangle$ (*the cups which are not blue*). Both of these descriptions satisfy all three constraints given above and distinguish c_1 and c_2 all other entities in the context.

Extending the algorithm with disjunctive properties. To take into account disjunctive properties, the constraints used can be modified as indicated follows:

$L_S = L_{S1} \vee \ldots \vee L_{Sn}$ is a distinguishing description for a set of individuals iff:

- $1 \leq n \leq |S|$
- $S = S_1 \cup \ldots \cup S_n$
- For $1 \leq i \leq n$, L_{Si} is a basic distinguishing description for S_i

That is, the algorithm looks for a tuple of sets such that the union $S_1 \cup \ldots \cup S_n$ of the tuple's elements is the target set S and such that for each set S_i in that tuple there is a basic distinguishing description L_{Si}. The resulting description is the disjunctive description $L_{S1} \vee \ldots \vee L_{Sn}$ where each L_{Si} is a conjunctive description of S_i. As before solutions are searched for in increasing order of size (i.e., of literals occurring in the description) by distributing over the cardinality of the resulting description.

Integration with surface realisation. To permit the generation of definite descriptions, the constraint-based algorithm for generating distinguishing descriptions presented above needs to be integrated with surface realisation. Assuming, as is usual, that the generation process is driven by communicative goals and, in particular, by informing and describing goals, this can be done by simply updating the current goal semantics with distinguishing descriptions. Whenever an entity must be described which is discourse old, a distinguishing description will first be computed for that entity using the above constraint solver and then added to the current goal semantics thereby driving further generation. Given some overall goal semantics, the generator then seeks to realise this goal semantics by building a phrase structure tree that (i) realises that goal semantics, (ii) is syntactically complete and (iii) is pragmatically appropriate.

Implementation. The constraint solver and the surface realisation algorithm sketched above have been implemented within the INDIGEN generator using the concurrent constraint programming language Mozart/Oz (Programming Systems Lab 1998) which supports set variables ranging over finite sets of integers and provides an efficient implementation of the associated constraint theory. The proof-of-concept implementation includes the constraint solver described above and its integration in a chart-based generator integrating surface realisation and inference. The integration of the constraint solver within the generator permits realizing definite NPs including negative information (*the cat that is not white*) and simple conjunctions (*The cat and the dog*).

5.2. Problems with incremental content determination

As argued in (Gardent 2002), the incremental algorithm, especially when generalised to boolean properties and sets of individuals, might yield cognitively and linguistically inadequate distinguishing descriptions.

Consider for instance a context such as pictured in Figure 9 and suppose the target set is $\{x_1, x_2, x_3, x_4\}$.

	white	*small*	*medium*	*big*	*dog*	*rabbit*	*cat*	*horse*	*sheep*
x_1	•	•			•				
x_2	•	•				•			
x_3				•			•		
x_4	•			•			•		
x_5	•			•					•
x_6	•		•						•
x_7			•						•
x_8			•					•	
x_9			•					•	
x_{10}			•					•	

Figure 9. The dog, the rabbit, and the cat.

Recall that to build a distinguishing description for a given target set, the incremental algorithm goes through the list of available properties in a given order and selects from it those properties which at each step (i) have the target set in their extension (all objects in the target set must have the selected property) and (ii) eliminate some distractor (the extension of the selected property may not be contained in the current distractor set).

The ordering of the properties is fixed in two ways. First, the disjunctive length is considered: the algorithm starts with disjunctive properties of length one, then goes on to disjunctive properties of length two etc. (cf. van Deemter 2001). Second, properties are ordered using sortal information (cf. Dale and Reiter 1995). For instance, the search through the available properties for the above example could be fixed to follow the order:

type < size < colour

Given these assumptions, the steps followed by the incremental algorithm to build a distinguishing description for the target set $\{x_1, x_2, x_3, x_4\}$ in the context given in Figure 9 might be as sketched in Figure 10.

First, a non-disjunctive type property is looked for whose extension contains the target set and is not contained in the current distractor set namely $\{x_5, x_6, x_7, x_8, x_9, x_{10}\}$. Two properties satisfy these criteria: ¬*horse* and ¬*sheep*. Suppose the property ¬*horse* is selected. It is then added to the distinguishing description (which is initially empty) and the distractor set is updated to the intersection of the current distractor set with the extension of the selected property namely, $\{x_5, x_6, x_7\}$.

		selected property	distractor set
disjunct of length 1			
	type	$\neg$ *horse*	$\{x_5, x_6, x_7\}$
	size	$\neg$ *medium*	$\{x_5\}$
	colour	No applicable property.	$\{x_5\}$
disjunct of length 2			
	type	No applicable property.	$\{x_5\}$
	size	No applicable property.	$\{x_5\}$
	colour	No applicable property.	$\{x_5\}$
disjunct of length 3			
	type	*dog* $\vee$ *rabbit* $\vee$ *cat*	$\{x_5\}$

Figure 10. Generating descriptions of sets with the incremental algorithm.

Next the "size properties" are considered and the property $\neg$*medium* is selected reducing the distractor set to $\{x_5\}$. No colour property satisfies the selection criteria hence disjunctive properties of length two are considered none of them satisfies the selection criteria. When considering disjunctive properties of length three, the disjunctive type property *dog* $\vee$ *rabbit* $\vee$ *cat* is selected thus yielding an empty distractor set. At this stage generation halts yielding the distinguishing description:

$$(\neg horse) \wedge (\neg medium) \wedge (dog \vee rabbit \vee cat)$$

That is, the incremental algorithm will in this case yield a distinguishing description which can be paraphrased as

the dogs, rabbits and cats that are not horses and that are not medium size

when a much shorter and more natural distinguishing description would in this case be the one paraphrased as

the dogs, the rabbits and the cats.

More generally, this example illustrates three types of problems for the incremental approach:

- **Contextually redundant descriptions:** the description produced might be contextually redundant in that a property present in the description might be entailed *in the given context* by some other information present elsewhere in the description. For instance the ($\neg$*medium*) property is contextually redundant in the above description as in the given context, dogs, rabbits and cats are all either big or small hence not medium size.

- **Epistemically redundant descriptions:** the description produced might be epistemically redundant in that a property it contains follows from some other information present elsewhere in the generated description and from our general knowledge about the world. For instance the (¬*horse*) property is epistemically redundant in the above description since we know that dogs, rabbits and cats cannot be horses.
- **Logically complex descriptions:** the description produced might be unnecessarily complex due to a high number of logical connectives. For instance, the description generated for the above example by the incremental algorithm contains two negations, two disjunctions and two conjunctions whilst a much simpler distinguishing description exists which contains only two disjunctions.

In contrast, because it produces minimal descriptions, the non-incremental algorithm for computing distinguishing descriptions presented in the previous section is not affected by any of these problems.

5.3. Problems with incremental surface realisation

In case of success, the output of the incremental algorithm is a distinguishing description, which is a conjunction of possibly complex properties whose denotation equals the target set.

To further realise this distinguishing description into a definite description, a kind of "double incrementality" has sometimes been proposed (Horacek 1997; Stone 1998) which consists in interleaving the incremental algorithm with surface realisation, i.e., with the construction of the syntactic tree associated by the grammar with the input semantic representation. Each selected property is used to retrieve a lexical entry whose semantics is this property and this lexical entry is immediately integrated into the tree which has been generated so far. In this way, it is ensured that the distinguishing description can be realised as a definite description (when the current tree cannot be updated with the selected lexical entry, another property is selected).

Thus in these doubly incremental approaches, it is not only the content but also the form which is determined greedily, i.e., locally and without backtracking. In other words, no global information is available which could

help in planning the definite NP. As we shall now see, this lack of global information may result in very unnatural definite NPs.

Consider again the above example. As noted, the distinguishing description produced by the incremental algorithm could in this case be

$$(\neg\ horse) \wedge (\neg\ medium) \wedge (dog \vee rabbit \vee cat)$$

If, as suggested by the doubly incremental algorithms, properties are realised in order of selection, the successive realisation phases will roughly be as follows:

the non horses
the non medium sized non horses
the non medium sized non horses that are either dogs, rabbits or cats

where, as noted above, a much more natural realisation of the input distinguishing description would be: *the dogs, rabbits and cats that are not horses and that are not medium size.*

The problem is that the order in which the incremental algorithm selects properties and the order in which properties can best be realised are governed by completely orthogonal constraints. More generally, the lack of global information concerning the semantics of the NP to be generated means that the overall structure of the NP cannot be optimised. So, for instance, the generator is in this case unable to recognise that the "best" definite description realising the computed distinguishing description is a three disjuncts NP with two conjoined modifiers (rather than a one disjunct NP with three disjoined modifiers).

In contrast, a surface realiser that is guided by a goal semantics (e.g., a distinguishing description) can use this semantics to plan and optimise the structure of the generated constituent (e.g., a definite description).

To start with, since the semantic information to be realised is given globally, realisation can be dictated by the grammar on the basis of the selected lexical entries and of their syntactic combinatorics rather than on the order in which properties are selected.

Further, various optimisation strategies can be devised based on the structure of the semantic input. Thus for instance, the surface realiser can detect from the form of the input delivered by the constraint solver described in the previous section (in essence a disjunctive normal form, i.e., a disjunction of conjunctions of literals) the number of conjuncts contained in the generated definite description – and recursively the number of disjuncts each conjunct should contain. Similarly, because negative and positive properties are kept separate, realisation strategies can be devised to optimise

the structure of the NP (if the distinguishing description contains several negative properties, for instance, antonyms can be searched for to minimise the number of negations). Finally, aggregation techniques could be applied to the input so as to avoid the repetition of semantic material appearing several times in the input distinguishing description.

A further advantage of the non-incremental algorithm presented in Section 5.1 is that it provides a richer input for surface realisation. Recall that the information output by the incremental algorithm focuses on property names rather than on property denotations. As a result, appropriate decisions regarding the singular/plural distinction cannot be made: since the cardinality of the extension of the property to be realised is unknown, it is not possible to decide whether the realised constituent should be plural (cardinality greater than one) or singular (cardinality equal to one).

In contrast, the input delivered by the non-incremental algorithm to the surface realisation component is a sequence of descriptions $L^{1\text{-}n}$ and of corresponding sets $S^{1\text{-}n}$ each description L_i in $L^{1\text{-}n}$ is a distinguishing description for the corresponding set S_i in $S^{1\text{-}n}$ and the sequence of descriptions $S^{1\text{-}n}$ is interpreted as a disjunction. This pairing of descriptions with sets, means that the correct number information can be determined: if a set has cardinality one, the corresponding NP will be singular; else the NP should be plural.

In sum, although the constraint based approach, not being incremental, does not allow for a tight interleaving of content planning and surface realisation and thus does not guarantee that the planned distinguishing description can actually be realised, it provides top-down guidance for the realiser which better supports the generation of syntactically optimal definite descriptions. Note further that the reverse problem holds for doubly incremental algorithms: although they guarantee that a definite description is built whenever a distinguishing description exists, they fail to guarantee that a referring expression is generated for those cases where no distinguishing description exists. Indeed in such cases, the doubly incremental algorithm will fail whereas the non-incremental algorithm will both detect the lack of distinguishing description and propose an alternative referring expression (e.g., an indefinite NP).

6. Conclusions

Starting from Dale and Reiter's (1995) incremental algorithm for generating definite descriptions, we have proposed two kinds of modifications to this algorithm.

First, we showed how the incremental algorithm can be interleaved with reasoning to support the generation of both indirectly co-referring definite descriptions and bridging descriptions. And second, we argued that non-incremental versions of the Dale and Reiter's algorithm should be explored both to compute distinguishing descriptions and to construct the associated definite descriptions. We proposed such an algorithm based on the use of set constraints and constraint programming.

These two extensions go in two different directions and address different problems, but, obviously, we will have to find a way to reconcile them and , e.g., extend the non-incremental constraint based algorithm to incorporate reasoning.

But also independently each of the approaches points to interesting questions concerning the generation of referring expressions. In our implementation of the non-incremental algorithm, we assumed that minimal solutions are the best solutions. This is, of course, an approximation that is not always true. It would be interesting to investigate and model the factors influencing the "quality" of a definite description. We believe that the constraint based approach we proposed provides us with an environment where we can experiment with different quality related restrictions.

The algorithm for the generation of inference based definite descriptions, on the other hand, touches on the issue of knowledge representation for natural language processing. It relies on the existence of knowledge bases specifying relations between concepts. In particular, the knowledge bases have to specify which concepts are linked by a relation that can be used as a bridging relation. To get a better understanding on what information has to be modelled in order to generate bridging descriptions, we undertook a corpus study. The results of this study show that the necessary knowledge is largely restricted to certain kinds of lexical knowledge, such as meronymy and verb frames. So, we are optimistic that, for a given domain, the knowledge bases necessary for the generation of bridging descriptions can be built. The next step would now be to actually build a larger knowledge base for some domain, investigating to what extend we can use existing resources (WordNet, FrameNet) and how we can do so in a systematic way. Another interesting question is whether the knowledge can be learned from corpora or

the web using, e.g., techniques as proposed by Bunescu (2003) or Markert, Nissim, and Modjeska (2003). This larger knowledge base could then also be used to test and evaluate our algorithm.

The corpus study has furthermore shown that the generation of referring expressions has to look at non-anaphoric uses of definite descriptions. They make up such a large percentage of all definite descriptions found in corpora that one cannot assume to generate natural sounding text without being able to properly account for this phenomenon. A second corpus study (Manuélian 2003) examining the distribution of given vs. new information in definite descriptions is a first step in this direction.

Notes

1. For a more detailed and more formal definition of intended and potential anchors see the technical report available at http://www.coli.uni-sb.de/~kris/papers/iwcs4-book.ps.gz.
2. The PAROLE corpus was created by the CNRS research unit ATILF (Analyse et Traitement Informatique de la Langue Française) and was made available to us in the context of a collaboration between ATILF and the LORIA research unit.

References

Baker, Collin F., Charles J. Fillmore, and John B. Lowe
1998 The Berkeley FrameNet project. *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and the 17th International Conference on Computational Linguistics*: 86–90.

Beaumont, Catherine, Josette Lecomte, and Nabil Hathout
1998 Etiquetage morpho-syntaxique du corpus "Le Monde" pour les besoins du projet PAROLE. Technical Report, INALF, Nancy.

Bunescu, Razvan
2003 Associative anaphora resolution: a web-based approach. *Proceedings of the EACL-Workshop: The Computational Treatment of Anaphora:* 47–52.

Clark, Herbert H.
1977 Bridging. In *Thinking: Readings in Cognitive Science.* P. N. Johnson-Laird and P. C. Wason (eds.), Cambridge University Press: 411–420.

Corblin, Francis
1987 *Indéfini, Défini et Démonstratif.* Genève/Paris: Librairie DROZ.
Corley, Steffan, Martin Corley, Frank Keller, Matthew W. Crocker, and Shari Trewin
2001 Finding syntactic structure in unparsed corpora: The Gsearch corpus query system. *Computers and the Humanities 35:* 81–94.
Dale, Robert
1992 *Generating Referring Expressions.* Cambridge, Massachussetts: MIT Press.
Dale, Robert and Ehud Reiter
1995 Computational Interpretation of the Gricean Maxims in the Generation of Referring Expressions. *Cognitive Science* 18: 233–263.
Deemter, Kees van
2002 Generation of Referring Expressions: Boolean Extensions of the Incremental Algorithm. *Computational Linguistics* 28 (1): 37–52.
Fellbaum, Christiane (ed.)
1998 *WordNet: An electronic lexical database*. Cambridge, Massachussetts: MIT Press.
Fraurud, Kari
1990 Definiteness and the processing of NPs in natural dicourse. *Journal of Semantics* 7: 395–433.
Gardent, Claire
2002 Generating minimal definite descriptions. *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*: 96–103.
Gardent, Claire, Hélène Manuélian, and Eric Kow
2003 Which bridges for bridging descriptions. *Proceedings of the EACL Workshop on Linguistically Interpreted Corpora:* 69–76.
Garey,Michael R. and David S. Johnson
1979 *Computers and Intractability: a Guide to the Theory of NP-Completeness*. San Francisco: W. H. Freeman.
Gundel, Jeanette K., Nancy Hedberg, and Ron Zacharski
1993 Cognitive status and the form of referring expressions in discourse. *Language* 69: 274–307.
Haarslev, Volker and Ralf Möller
2001 RACER system description. *Proceedings of the International Joint Conference on Automated Reasoning:* 701–705.

Horacek, Helmut
1997 An algorithm for generating referential descriptions with flexible interfaces. *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics:* 206–213.

Kleiber, Georges
1997 Des anaphors associatives méronymique aux anaphors associatives locatives. *Verbum* 19(1-2): 25–66.

Krahmer, Emiel and Mariët Theune
2001 Efficient context-sensitive generation of referring expressions. In *Information Sharing: Givenness and Newness in Language Processing*, Kees van Deemter and Roger Kibble (eds.). CSLI Publications: 223–264.

Lecomte, Josette
1997 Codage MULTEXT-GRACE pour l'action GRACE. Technical Report, INALF, Nancy.

Manuélian, Hélène
2003 Coreferential definite and demonstrative descriptions in French: a corpus study for generation. *Proceedings of the 8th ESSLLI Student Session:* 169–180.

Markert, Katja, Malvina Nissim, and Natalia Modjeska
2003 Using the web for nominal anaphora resolution. *Proceedings of the EACL-Workshop: The Computational Treatment of Anaphora:* 39–46.

Müller, Christoph and Michael Strube
2001 MMAX: A tool for the annotation of multi-modal corpora. *Proceedings of the 2nd IJCAI Workshop on Knowledge and Reasoning in Practical Dialogue Systems:* 45–50.

Poesio, Massimo and Renata Vieira
1998 A corpus-based investigation of definite descriptions use. *Computational Linguistics* 24(2): 183–216.

Programming Systems Lab Saarbrücken
1998 Mozart/Oz Webpage: http://www.mozart-oz.org.

Roy, Peter Van and Seif Haridi
2003 *Concepts, Techniques, and Models of Computer Programming*. Webpage: http://www.info.ucl.ac.be/people/PVR/book.html. To be published by MIT Press.

Stone, Matthew
1998 Modality in dialogue: planning, pragmatics and computation. Ph.D. thesis, Department of Computer & Information Science, University of Pennsylvania.

Winston, Morton E., Roger Chaffin, and Douglas Hermann
1987 A taxonomy of part-whole relations. *Cognitive Science* 11: 417–444.